D 16

```
FFFFFFFFFFFFFFF         111               111            AAAAAAAAA
FFFFFFFFFFFFFFF         111               111            AAAAAAAAA
FFFFFFFFFFFFFFF         111               111            AAAAAAAAA
FFF                  111111            111111         AAA         AAA
FFF                  111111            111111         AAA         AAA
FFF                  111111            111111         AAA         AAA
FFF                     111               111         AAA         AAA
FFF                     111               111         AAA         AAA
FFF                     111               111         AAA         AAA
FFFFFFFFFFFF            111               111         AAA         AAA
FFFFFFFFFFFF            111               111         AAA         AAA
FFFFFFFFFFFF            111               111         AAA         AAA
FFF                     111               111         AAAAAAAAAAAAAAA
FFF                     111               111         AAAAAAAAAAAAAAA
FFF                     111               111         AAAAAAAAAAAAAAA
FFF                     111               111         AAA         AAA
FFF                     111               111         AAA         AAA
FFF                     111               111         AAA         AAA
FFF                  111111111         111111111      AAA         AAA
FFF                  111111111         111111111      AAA         AAA
FFF                  111111111         111111111      AAA         AAA
```

```
                                   N  6
**FILE**ID**EXTDIR




EEEEEEEEEE  XX      XX  TTTTTTTTTT  DDDDDDDD      IIIIII    RRRRRRRR
EEEEEEEEEE  XX      XX  TTTTTTTTTT  DDDDDDDD      IIIIII    RRRRRRRR
EE          XX      XX      TT      DD      DD      II      RR      RR
EE          XX      XX      TT      DD      DD      II      RR      RR
EE            XX  XX        TT      DD      DD      II      RR      RR
EE            XX  XX        TT      DD      DD      II      RR      RR
EEEEEEEE        XX          TT      DD      DD      II      RRRRRRRR
EEEEEEEE        XX          TT      DD      DD      II      RRRRRRRR
EE            XX  XX        TT      DD      DD      II      RR  RR
EE            XX  XX        TT      DD      DD      II      RR  RR
EE          XX      XX      TT      DD      DD      II      RR    RR        ....
EE          XX      XX      TT      DD      DD      II      RR    RR        ....
EEEEEEEEEE  XX      XX      TT      DDDDDDDD      IIIIII    RR      RR      ....
EEEEEEEEEE  XX      XX      TT      DDDDDDDD      IIIIII    RR      RR      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0 MODULE EXTDIR (
0                       NOSAFE,
0                       LANGUAGE (BLISS32),
0                       IDENT = 'V04-000'
0                       ) =
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 !
1 ! FACILITY:  F11ACP Structure Level 1
1 !
1 ! ABSTRACT:
1 !
1 !       This routine extends a directory file.
1 !
1 ! ENVIRONMENT:
1 !
1 !       STARLET operating system, including privileged system services
1 !       and internal exec routines.
1 !
1 !--
1 !
1 !
1 ! AUTHOR:  Andrew C. Goldstein,  CREATION DATE:  15-Apr-1977  13:25
1 !
1 ! MODIFIED BY:
1 !
1 !       A0101   ACG0121         Andrew C. Goldstein,    16-Jan-1980  22:57
1 !               Make context save and restore into subroutines
1 !
1 !       A0100   ACG00001        Andrew C. Goldstein,  10-Oct-1978  20:02
1 !               Previous revision history moved to F11A.REV
1 !**
```

```
;   58          0058  1
;   59          0059  1
;   60          0060  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;   61          0061  1 REQUIRE 'SRC$:FCPDEF.B32';
;   62          0376  1
;   63          0377  1
;   64          0378  1 FORWARD ROUTINE
;   65          0379  1         EXTEND_DIR,                             ! extend directory file
;   66          0380  1         HANDLER;                                ! local condition handler
;    68          0381   1 GLOBAL ROUTINE EXTEND_DIR =
;    69          0382   1
;    70          0383   1 !++
;    71          0384   1 !
;    72          0385   1 ! FUNCTIONAL DESCRIPTION:
;    73          0386   1 !
;    74          0387   1 !       This routine extends a directory file. If allocated but unused space is
;    75          0388   1 !       present, this means simply pushing back the EOF and materializing a
;    76          0389   1 !       block of zeroes. If the file is to be physically extended, it is
;    77          0390   1 !       copied to a new location on the disk to keep it contiguous. Note
;    78          0391   1 !       this routine will not do a non-contiguous extend if the above fails,
;    79          0392   1 !       nor does it implement the automatic conversion of non-contiguous
;    80          0393   1 !       directories found in RSX-11. The frequency of use of these is deemed
;    81          0394   1 !       too low to be worthwhile.
;    82          0395   1 !
;    83          0396   1 ! CALLING SEQUENCE:
;    84          0397   1 !       EXTEND_DIR ()
;    85          0398   1 !
;    86          0399   1 ! INPUT PARAMETERS:
;    87          0400   1 !       NONE
;    88          0401   1 !
;    89          0402   1 ! IMPLICIT INPUTS:
;    90          0403   1 !       DIR_FCB: FCB of directory file
;    91          0404   1 !       DIR_WINDOW: window of directory
;    92          0405   1 !
;    93          0406   1 ! OUTPUT PARAMETERS:
;    94          0407   1 !       NONE
;    95          0408   1 !
;    96          0409   1 ! IMPLICIT OUTPUTS:
;    97          0410   1 !       DIR_RECORD: record number of first available record created
;    98          0411   1 !
;    99          0412   1 ! ROUTINE VALUE:
;   100          0413   1 !       address of next directory record to use
;   101          0414   1 !
;   102          0415   1 ! SIDE EFFECTS:
;   103          0416   1 !       directory extended, storage map altered, directory FCB & windows altered
;   104          0417   1 !
;   105          0418   1 !--
;   106          0419   1
;   107          0420   2 BEGIN
;   108          0421   2
;   109          0422   2 BUILTIN
;   110          0423   2       FP;
;   111          0424   2
;   112          0425   2 LOCAL
;   113          0426   2       FIB             : REF BBLOCK,   ! address of FIB for this operation
;   114          0427   2       FCB             : REF BBLOCK,   ! address of FCB for directory
;   115          0428   2       HEADER          : REF BBLOCK,   ! address of directory file header
;   116          0429   2       NEXT_VBN,                       ! next directory VBN to use
;   117          0430   2       NEW_SIZE,                       ! size to extend directory to
;   118          0431   2       NEW_LBN,                        ! starting LBN of new space
;   119          0432   2       LBN,                            ! current LBN in copy
;   120          0433   2       BUFFER,                         ! buffer address of current directory block
;   121          0434   2       MAP_AREA        : REF BBLOCK,   ! address of file header map area
;   122          0435   2       MAP_POINTER     : REF BBLOCK,   ! pointer to current retrieval pointer
;   123          0436   2       NEXT_LBN;                       ! LBN of next block to use
;   124          0437   2
;  125          0438   2 EXTERNAL
;  126          0439   2          LOCAL_FIB          : BBLOCK,          ! FIB for main file operation
;  127          0440   2          SECOND_FIB         : BBLOCK,          ! FIB for secondary operations
;  128          0441   2          DIR_FCB            : REF BBLOCK,      ! address of directory FCB
;  129          0442   2          DIR_WINDOW         : REF BBLOCK,      ! address of directory window
;  130          0443   2          PRIMARY_FCB        : REF BBLOCK,      ! FCB of file in process
;  131          0444   2          UNREC_COUNT,                          ! count of unrecorded blocks
;  132          0445   2          UNREC_LBN,                            ! starting LBN of unrecorded blocks
;  133          0446   2          DIR_RECORD;                           ! record number of directory entry
;  134          0447   2
;  135          0448   2 EXTERNAL ROUTINE
;  136          0449   2          SAVE_CONTEXT,                         ! save reentrant context area
;  137          0450   2          RESTORE_CONTEXT,                      ! restore reentrant context area
;  138          0451   2          READ_HEADER,                          ! read file header
;  139          0452   2          ALLOC_BLOCKS,                         ! allocate blocks from storage map
;  140          0453   2          RETURN_BLOCKS,                        ! return blocks to storage map
;  141          0454   2          MAP_VBN,                              ! map virtual to logical
;  142          0455   2          READ_BLOCK,                           ! read a disk block
;  143          0456   2          RESET_LBN,                            ! assign new LBN to buffer
;  144          0457   2          WRITE_BLOCK,                          ! write block to disk
;  145          0458   2          CREATE_BLOCK,                         ! fabricate a block buffer
;  146          0459   2          INVALIDATE,                           ! invalidate a buffer
;  147          0460   2          TRUNCATE_HEADER,                      ! truncate file header
;  148          0461   2          CHECKSUM,                             ! compute file header checksum
;  149          0462   2          WRITE_HEADER,                         ! write file header
;  150          0463   2          INIT_FCB,                             ! update file control block
;  151          0464   2          ZERO_WINDOWS;                         ! invalidate related file windows
;  152          0465   2
;  153          0466   2
;  154          0467   2 ! First save the current context, since this is a secondary file operation.
;  155          0468   2 ! Set up the secondary context pointers. Then read the directory file header.
;  156          0469   2 !
;  157          0470   2
;  158          0471   2 SAVE_CONTEXT ();
;  159          0472   2 PRIMARY_FCB = FCB = .DIR_FCB;
;  160          0473   2 FIB = SECOND_FIB;
;  161          0474   2 CH$MOVE (FIB$S_FID, LOCAL_FIB[FIB$W_DID], FIB[FIB$W_FID]);
;  162          0475   2
;  163          0476   2 HEADER = READ_HEADER (0, .FCB);
;  164          0477   2
;  165          0478   2 ! The next VBN to use is the current directory eof block number. If the block
;  166          0479   2 ! is not present in the file, the directory must be physically extended.
;  167          0480   2 !
;  168          0481   2
;  169          0482   2 NEXT_VBN = .FCB[FCB$L_EFBLK] + 1;
;  170          0483   2
;  171          0484   2 IF .NEXT_VBN GTRU .FCB[FCB$L_FILESIZE]
;  172          0485   2 THEN
;  173          0486   3     BEGIN
;  174          0487   3
;  175          0488   3 ! Compute the number of blocks needed (50% of the current directory size)
;  176          0489   3 ! and allocate the new space contiguously. Limit the number of blocks
;  177          0490   3 ! allocated to what will fit in the map area of the header.
;  178          0491   3 !
;  179          0492   3
;  180          0493   3     MAP_AREA = .HEADER + .HEADER[FH1$B_MPOFFSET]*2;
;  181          0494   3     NEW_SIZE = .FCB[FCB$L_FILESIZE] + MAXU (.FCB[FCB$L_FILESIZE]/2, 1);
;  182        0495  3       IF .FCB[FCB$L_FILESIZE] GEQU 2048 THEN ERR_EXIT (SS$_DIRFULL);
;  183        0496  3       IF .NEW_SIZE GTRU 2048 THEN NEW_SIZE = 2048;
;  184        0497  3
;  185        0498  3       FIB[FIB$V_ALCON] = 1;
;  186        0499  3       FIB[FIB$V_FILCON] = 1;
;  187        0500  3       ALLOC_BLOCKS (.FIB, .NEW_SIZE, NEW_LBN, NEW_SIZE);
;  188        0501  3       UNREC_COUNT = .NEW_SIZE;
;  189        0502  3       UNREC_LBN = .NEW_LBN;
;  190        0503  3
;  191        0504  3   ! Now copy the directory data from the old directory file to the newly
;  192        0505  3   ! allocated space.
;  193        0506  3   !
;  194        0507  3
;  195        0508  3       INCR VBN FROM 1 TO .FCB[FCB$L_FILESIZE] DO
;  196        0509  4           BEGIN
;  197        0510  4           IF .FCB[FCB$L_STLBN] NEQ 0
;  198        0511  4           THEN LBN = .VBN + .FCB[FCB$L_STLBN] - 1
;  199        0512  4           ELSE LBN = MAP_VBN (.VBN, .DIR_WINDOW);
;  200        0513  4
;  201        0514  4           BUFFER = READ_BLOCK (.LBN, 1, DIRECTORY_TYPE);
;  202        0515  4           RESET_LBN (.BUFFER, .VBN + .NEW_LBN - 1);
;  203        0516  4           WRITE_BLOCK (.BUFFER);
;  204        0517  3           END;
;  205        0518  3
;  206        0519  3   ! Now deallocate the old directory blocks. Then build retrieval pointers
;  207        0520  3   ! for the new blocks in the file header. Do the truncation with a local
;  208        0521  3   ! condition handler enabled for special error recovery.
;  209        0522  3   !
;  210        0523  3
;  211        0524  3       .FP = HANDLER;
;  212        0525  3       TRUNCATE_HEADER (.FIB, .HEADER, DEALLOC_BLOCKS);
;  213        0526  3       .FP = 0;
;  214        0527  3
;  215        0528  3       MAP_POINTER = .MAP_AREA + FM1$C_POINTERS;
;  216        0529  3
;  217        0530  3       DO
;  218        0531  4           BEGIN
;  219        0532  4           MAP_AREA[FM1$B_INUSE] = .MAP_AREA[FM1$B_INUSE] + 2;
;  220        0533  4
;  221        0534  4           MAP_POINTER[FM1$B_HIGHLBN] = .NEW_LBN<16,8>;
;  222        0535  4           MAP_POINTER[FM1$B_COUNT] = MIN (.NEW_SIZE, 256) - 1;
;  223        0536  4           MAP_POINTER[FM1$W_LOWLBN] = .NEW_LBN<0,16>;
;  224        0537  4           MAP_POINTER = .MAP_POINTER + 4;
;  225        0538  4
;  226        0539  4           NEW_LBN = .NEW_LBN + MIN (.NEW_SIZE, 256);
;  227        0540  4           NEW_SIZE = .NEW_SIZE - MIN (.NEW_SIZE, 256);
;  228        0541  4           END
;  229        0542  3       UNTIL .NEW_SIZE EQL 0;
;  230        0543  3
;  231        0544  3       UNREC_COUNT = 0;
;  232        0545  3       HEADER[FH1$V_CONTIG] = 1;                   ! mark file contiguous
;  233        0546  3       KERNEL_CALL (ZERO_WINDOWS, .FCB);
;  234        0547  2       END;                                        ! end of directory extension
;  235        0548  2
;  236        0549  2   ! Now that we have enough space in the directory, push the end of file
;  237        0550  2   ! mark back one block and materialize the new block in memory. Also
;  238        0551  2   ! update the FCB and flush any windows on it.
;  239         0552  2 !
;  240         0553  2
;  241         0554  2 BBLOCK [HEADER[FH1$W_RECATTR], FAT$W_EFBLKL] = .NEXT_VBN + 1;
;  242         0555  2 BBLOCK [HEADER[FH1$W_RECATTR], FAT$W_FFBYTE] = 0;
;  243         0556  2 KERNEL_CALL (INIT_FCB, .FCB, .HEADER);
;  244         0557  2 BBLOCK [HEADER[FH1$W_RECATTR], FAT$W_HIBLKL] = .FCB[FCB$L_FILESIZE];
;  245         0558  2 CHECKSUM (.HEADER);
;  246         0559  2 WRITE_HEADER ();
;  247         0560  2
;  248         0561  3 NEXT_LBN = (
;  249         0562  3      IF .FCB[FCB$L_STLBN] NEQ 0
;  250         0563  3      THEN .FCB[FCB$L_STLBN] + .NEXT_VBN - 1
;  251         0564  3      ELSE MAP_VBN (.NEXT_VBN, .DIR_WINDOW)
;  252         0565  2      );
;  253         0566  2 BUFFER = CREATE_BLOCK (.NEXT_LBN, 1, DIRECTORY_TYPE);
;  254         0567  2 DIR_RECORD = (.NEXT_VBN - 1) * 32 + 1;
;  255         0568  2
;  256         0569  2 ! Finally switch back to primary context.
;  257         0570  2 !
;  258         0571  2
;  259         0572  2 RESTORE_CONTEXT ();
;  260         0573  2
;  261         0574  2 RETURN .BUFFER;
;  262         0575  2
;  263         0576  1 END;                                           ! end of routine EXTEND_DIR

                                                                   .TITLE  EXTDIR
                                                                   .IDENT  \V04-000\

                                                                   .EXTRN  LOCAL_FIB, SECOND_FIB
                                                                   .EXTRN  DIR_FCB, DIR_WINDOW
                                                                   .EXTRN  PRIMARY_FCB, UNREC_COUNT
                                                                   .EXTRN  UNREC_LBN, DIR_RECORD
                                                                   .EXTRN  SAVE_CONTEXT, RESTORE_CONTEXT
                                                                   .EXTRN  READ_HEADER, ALLOC_BLOCKS
                                                                   .EXTRN  RETURN_BLOCKS, MAP_VBN
                                                                   .EXTRN  READ_BLOCK, RESET_LBN
                                                                   .EXTRN  WRITE_BLOCK, CREATE_BLOCK
                                                                   .EXTRN  INVALIDATE, TRUNCATE_HEADER
                                                                   .EXTRN  CHECKSUM, WRITE_HEADER
                                                                   .EXTRN  INIT_FCB, ZERO_WINDOWS
                                                                   .EXTRN  SYS$CMKRNL

                                                                   .PSECT  $CODE$,NOWRT,2

                                                   0FFC 00000      .ENTRY  EXTEND_DIR, Save R2,R3,R4,R5,R6,R7,R8,R9,-   ; 0381
                                                                           R10,R11                                       ;
                        5B 00000000G    9F   9E 00002      MOVAB   @#SYS$CMKRNL, R11                                      ;
                        5E              08   C2 00009      SUBL2   #8, SP                                                 ;
              0000G     CF              00   FB 0000C      CALLS   #0, SAVE_CONTEXT                                       ; 0471
                        56      0000G   CF   D0 00011      MOVL    DIR_FCB, FCB                                           ; 0472
              0000G     CF              56   D0 00016      MOVL    FCB, PRIMARY_FCB                                       ;
                        57      0000G   CF   9E 0001B      MOVAB   SECOND_FIB, FIB                                        ; 0473
  04     A7   0000G     CF              06   28 00020      MOVC3   #6, LOCAL_FIB+10, 4(FIB)                               ; 0474
                                        56   DD 00027      PUSHL   FCB                                                    ; 0476
                                        7E   D4 00029      CLRL    -(SP)                                                  ;
```

```
           0000G  CF              02  FB 0002B          CALLS   #2, READ_HEADER                     ;
                  54              50  D0 00030          MOVL    R0, HEADER                          ;
  53          3C  A6              01  C1 00033          ADDL3   #1, 60(FCB), NEXT_VBN               ; 0482
              38  A6              53  D1 00038          CMPL    NEXT_VBN, 56(FCB)                   ; 0484
                                  03  1A 0003C          BGTRU   1$                                  ;
                                010E  31 0003E          BRW     12$                                 ;
                  50        01    A4  9A 00041 1$:      MOVZBL  1(HEADER), R0                       ; 0493
                  52            6440  3E 00045          MOVAW   (HEADER)[R0], MAP_AREA              ;
  50          38  A6              02  C7 00049          DIVL3   #2, 56(FCB), R0                     ; 0494
                                  03  12 0004E          BNEQ    2$                                  ;
                  50              01  D0 00050          MOVL    #1, R0                              ;
                  6E        38  B640  9E 00053 2$:      MOVAB   @56(FCB)[R0], NEW_SIZE              ;
        00000800  8F        38    A6  D1 00058          CMPL    56(FCB), #2048                      ; 0495
                                  05  1F 00060          BLSSU   3$                                  ;
                          0860    8F  BF 00062          CHMU    #2144                               ;
                                      04 00066          RET                                         ;
        00000800  8F              6E  D1 00067 3$:      CMPL    NEW_SIZE, #2048                     ; 0496
                                  05  1B 0006E          BLEQU   4$                                  ;
                  6E      0800    8F  3C 00070          MOVZWL  #2048, NEW_SIZE                     ;
              16  A7              05  88 00075 4$:      BISB2   #5, 22(FIB)                         ; 0499
                                  5E  DD 00079          PUSHL   SP                                  ; 0500
                            08    AE  9F 0007B          PUSHAB  NEW_LBN                             ;
                            08    AE  DD 0007E          PUSHL   NEW_SIZE                            ;
                                  57  DD 00081          PUSHL   FIB                                 ;
           0000G  CF              04  FB 00083          CALLS   #4, ALLOC_BLOCKS                    ;
           0000G  CF              6E  D0 00088          MOVL    NEW_SIZE, UNREC_COUNT               ; 0501
           0000G  CF        04    AE  D0 0008D          MOVL    NEW_LBN, UNREC_LBN                  ; 0502
                  58        38    A6  D0 00093          MOVL    56(FCB), R8                         ; 0508
                                  55  D4 00097          CLRL    VBN                                 ;
                                  42  11 00099          BRB     8$                                  ;
                            30    A6  D5 0009B 5$:      TSTL    48(FCB)                             ; 0510
                                  0B  13 0009E          BEQL    6$                                  ;
  50              55        30    A6  C1 000A0          ADDL3   48(FCB), VBN, R0                    ; 0511
                  59        FF    A0  9E 000A5          MOVAB   -1(R0), LBN                         ;
                                  0E  11 000A9          BRB     7$                                  ;
                         0000G    CF  DD 000AB 6$:      PUSHL   DIR_WINDOW                          ; 0512
                                  55  DD 000AF          PUSHL   VBN                                 ;
           0000G  CF              02  FB 000B1          CALLS   #2, MAP_VBN                         ;
                  59              50  D0 000B6          MOVL    R0, LBN                             ;
                                  02  DD 000B9 7$:      PUSHL   #2                                  ; 0514
                                  01  DD 000BB          PUSHL   #1                                  ;
                                  59  DD 000BD          PUSHL   LBN                                 ;
           0000G  CF              03  FB 000BF          CALLS   #3, READ_BLOCK                      ;
                  5A              50  D0 000C4          MOVL    R0, BUFFER                          ;
  50              55        04    AE  C1 000C7          ADDL3   NEW_LBN, VBN, R0                    ; 0515
                            FF    A0  9F 000CC          PUSHAB  -1(R0)                              ;
                                  5A  DD 000CF          PUSHL   BUFFER                              ;
           0000G  CF              02  FB 000D1          CALLS   #2, RESET_LBN                       ;
                                  5A  DD 000D6          PUSHL   BUFFER                              ; 0516
           0000G  CF              01  FB 000D8          CALLS   #1, WRITE_BLOCK                     ;
  BA              55              58  F3 000DD 8$:      AOBLEQ  R8, VBN, 5$                         ; 0508
                  6D     0000V    CF  9E 000E1          MOVAB   HANDLER, (FP)                       ; 0524
                                  01  DD 000E6          PUSHL   #1                                  ; 0525
                                  54  DD 000E8          PUSHL   HEADER                              ;
                                  57  DD 000EA          PUSHL   FIB                                 ;
           0000G  CF              03  FB 000EC          CALLS   #3, TRUNCATE_HEADER                 ;
                                  6D  D4 000F1          CLRL    (FP)                                ; 0526
```

```
                     50       0A  A2 9E 000F3       MOVAB    10(R2), MAP_POINTER                     ; 0528
                 08  A2           02 80 000F7 9$:   ADDB2    #2, 8(MAP_AREA)                         ; 0532
                     60       06  AE 90 000FB       MOVB     NEW_LBN+2, (MAP_POINTER)                ; 0534
                     51           6E D0 000FF       MOVL     NEW_SIZE, R1                            ; 0535
           00000100  8F           51 D1 00102       CMPL     R1, #256                                ;
                                  05 15 00109       BLEQ     10$                                     ;
                     51     0100  8F 3C 0010B       MOVZWL   #256, R1                                ;
01    A0             51           01 83 00110 10$:  SUBB3    #1, R1, 1(MAP_POINTER)                  ;
                 02  A0       04  AE B0 00115       MOVW     NEW_LBN, 2(MAP_POINTER)                 ; 0536
                     50           04 C0 0011A       ADDL2    #4, MAP_POINTER                         ; 0537
                     51           6E D0 0011D       MOVL     NEW_SIZE, R1                            ; 0539
           00000100  8F           51 D1 00120       CMPL     R1, #256                                ;
                                  05 15 00127       BLEQ     11$                                     ;
                     51     0100  8F 3C 00129       MOVZWL   #256, R1                                ;
                 04  AE           51 C0 0012E 11$:  ADDL2    R1, NEW_LBN                             ;
                     6E           51 C2 00132       SUBL2    R1, NEW_SIZE                            ; 0540
                                  6E D5 00135       TSTL     NEW_SIZE                                ; 0542
                                  BE 12 00137       BNEQ     9$                                      ;
                           0000G  CF D4 00139       CLRL     UNREC_COUNT                             ; 0544
                 0C  A4       80  8F 88 0013D       BISB2    #128, 12(HEADER)                        ; 0545
                                  56 DD 00142       PUSHL    FCB                                     ; 0546
                                  01 DD 00144       PUSHL    #1                                      ;
                                  5E DD 00146       PUSHL    SP                                      ;
                           0000G  CF 9F 00148       PUSHAB   ZERO_WINDOWS                            ;
                     6B           04 FB 0014C       CALLS    #4, SYS$CMKRNL                          ;
18    A4             53           01 A1 0014F 12$:  ADDW3    #1, NEXT_VBN, 24(HEADER)                ; 0554
                              1A  A4 B4 00154       CLRW     26(HEADER)                              ; 0555
                                  54 DD 00157       PUSHL    HEADER                                  ; 0556
                                  56 DD 00159       PUSHL    FCB                                     ;
                                  02 DD 0015B       PUSHL    #2                                      ;
                                  5E DD 0015D       PUSHL    SP                                      ;
                           0000G  CF 9F 0015F       PUSHAB   INIT_FCB                                ;
                     6B           05 FB 00163       CALLS    #5, SYS$CMKRNL                          ;
                 14  A4       38  A6 B0 00166       MOVW     56(FCB), 20(HEADER)                     ; 0557
                                  54 DD 0016B       PUSHL    HEADER                                  ; 0558
              0000G  CF           01 FB 0016D       CALLS    #1, CHECKSUM                            ;
              0000G  CF           00 FB 00172       CALLS    #0, WRITE_HEADER                        ; 0559
                              30  A6 D5 00177       TSTL     48(FCB)                                 ; 0562
                                  09 13 0017A       BEQL     13$                                     ;
      50             53       30  A6 C1 0017C       ADDL3    48(FCB), NEXT_VBN, R0                   ; 0563
                                  50 D7 00181       DECL     NEXT_LBN                                ;
                                  0B 11 00183       BRB      14$                                     ;
                           0000G  CF DD 00185 13$:  PUSHL    DIR_WINDOW                              ; 0564
                                  53 DD 00189       PUSHL    NEXT_VBN                                ;
              0000G  CF           02 FB 0018B       CALLS    #2, MAP_VBN                             ;
                                  02 DD 00190 14$:  PUSHL    #2                                      ; 0566
                                  01 DD 00192       PUSHL    #1                                      ;
                                  50 DD 00194       PUSHL    NEXT_LBN                                ;
              0000G  CF           03 FB 00196       CALLS    #3, CREATE_BLOCK                        ;
                     5A           50 D0 0019B       MOVL     R0, BUFFER                              ;
                     53           20 C4 0019E       MULL2    #32, R3                                 ; 0567
              0000G  CF       E1  A3 9E 001A1       MOVAB    -31(R3), DIR_RECORD                     ;
              0000G  CF           00 FB 001A7       CALLS    #0, RESTORE_CONTEXT                     ; 0572
                     50           5A D0 001AC       MOVL     BUFFER, R0                              ; 0574
                                     04 001AF       RET                                              ; 0576

; Routine Size:  432 bytes,    Routine Base:  $CODE$ + 0000
```

```
;   265         0577    1 ROUTINE HANDLER (SIGNAL, MECHANISM) =
;   266         0578    1
;   267         0579    1 !++
;   268         0580    1 !
;   269         0581    1 ! FUNCTIONAL DESCRIPTION:
;   270         0582    1 !
;   271         0583    1 !       This routine is the condition handler for directory extension. It is
;   272         0584    1 !       enabled only during the truncate call (deallocating the old directory
;   273         0585    1 !       blocks). Normal error handling would cause the entire directory to
;   274         0586    1 !       be dropped on the floor. Since we already have a new good copy, we
;   275         0587    1 !       should forge ahead. Note that no error status is returned to the user,
;   276         0588    1 !       although we will log a system error.
;   277         0589    1 !
;   278         0590    1 !
;   279         0591    1 ! CALLING SEQUENCE:
;   280         0592    1 !       HANDLER (ARG1, ARG2)
;   281         0593    1 !
;   282         0594    1 ! INPUT PARAMETERS:
;   283         0595    1 !       ARG1: address of signal array
;   284         0596    1 !       ARG2: address of mechanism array
;   285         0597    1 !
;   286         0598    1 ! IMPLICIT INPUTS:
;   287         0599    1 !       FILE_HEADER: address of directory file header
;   288         0600    1 !
;   289         0601    1 ! OUTPUT PARAMETERS:
;   290         0602    1 !       NONE
;   291         0603    1 !
;   292         0604    1 ! IMPLICIT OUTPUTS:
;   293         0605    1 !       NONE
;   294         0606    1 !
;   295         0607    1 ! ROUTINE VALUE:
;   296         0608    1 !       SS$_RESIGNAL or none if unwind
;   297         0609    1 !
;   298         0610    1 ! SIDE EFFECTS:
;   299         0611    1 !       file header map area cleaned out
;   300         0612    1 !
;   301         0613    1 !--
;   302         0614    1
;   303         0615    2 BEGIN
;   304         0616    2
;   305         0617    2 MAP
;   306         0618    2       SIGNAL          : REF BBLOCK,   ! signal array arg
;   307         0619    2       MECHANISM       : REF BBLOCK;   ! mechanism array arg
;   308         0620    2
;   309         0621    2 LOCAL
;   310         0622    2       MAP_AREA        : REF BBLOCK;   ! address of header map area
;   311         0623    2
;   312         0624    2 EXTERNAL
;   313         0625    2       FILE_HEADER     : REF BBLOCK;   ! address of directory file header
;   314         0626    2
;   315         0627    2 EXTERNAL ROUTINE
;   316         0628    2       SYS$UNWIND      : ADDRESSING_MODE (ABSOLUTE);
;   317         0629    2                                       ! system unwind service
;   318         0630    2
;   319         0631    2
;   320         0632    2 ! Check the condition code for FCP error exit and check that it is not a
;   321         0633    2 ! write error. Then initialize the header's map area and unwind. On other
;  322          0634  2 ! signals we simply resignal.
;  323          0635  2 !
;  324          0636  2
;  325          0637  2 IF .SIGNAL[CHF$L_SIG_NAME] EQL SS$_CMODUSER
;  326          0638  2 THEN
;  327          0639  3     BEGIN
;  328          0640  3     MAP_AREA = .FILE_HEADER + .FILE_HEADER[FH1$B_MPOFFSET]*2;
;  329          0641  3     MAP_AREA[FM1$B_INUSE] = 0;
;  330          0642  3     CH$FILL (0, .MAP_AREA[FM1$B_AVAIL]*2, .MAP_AREA + FM1$C_POINTERS);
;  331          0643  3
;  332          0644  3     SYS$UNWIND (MECHANISM[CHF$L_MCH_DEPTH], 0);
;  333          0645  2     END;
;  334          0646  2
;  335          0647  2 RETURN SS$_RESIGNAL;                         ! status is irrelevant if unwind
;  336          0648  2
;  337          0649  1 END;                                         ! end of routine handler



                                                                        .EXTRN  FILE_HEADER, SYS$UNWIND

                                                  003C 00000 HANDLER:.WORD    Save R2,R3,R4,R5                              ; 0577
                            50       04     AC     D0 00002          MOVL     SIGNAL, R0                                    ; 0637
                 00000424   8F       04     A0     D1 00006          CMPL     4(R0), #1060                                  ;
                                            2C     12 0000E          BNEQ     1$                                            ;
                            51     0000G    CF     D0 00010          MOVL     FILE_HEADER, R1                               ; 0640
                            50       01     A1     9A 00015          MOVZBL   1(R1), R0                                     ;
                            50          6140       3E 00019          MOVAW    (R1)[R0], MAP_AREA                            ;
                                     08     A0     94 0001D          CLRB     8(MAP_AREA)                                   ; 0641
                            51       09     A0     9A 00020          MOVZBL   9(MAP_AREA), R1                               ; 0642
                            51              02     C4 00024          MULL2    #2, R1                                        ;
        51         00       6E              00     2C 00027          MOVC5    #0, (SP), #0, R1, 10(MAP_AREA)                ;
                                     0A     A0        0002C                                                                 ;
                                            7E     D4 0002E          CLRL     -(SP)                                         ; 0644
                    7E       08     AC      08     C1 00030          ADDL3    #8, MECHANISM, -(SP)                          ;
                 00000000G   9F             02     FB 00035          CALLS    #2, @#SYS$UNWIND                              ;
                            50     0918     8F     3C 0003C 1$:      MOVZWL   #2328, R0                                     ; 0647
                                                   04 00041          RET                                                    ; 0649

; Routine Size:  66 bytes,    Routine Base:  $CODE$ + 01B0


;  338          0650  1
;  339          0651  1 END
;  340          0652  0 ELUDOM




;                                 PSECT SUMMARY
;
;       Name                  Bytes                            Attributes
;
;   $CODE$                          498  NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)
;                        Library Statistics
;
;                                                  --------- Symbols --------       Pages        Processing
;        File                                      Total     Loaded   Percent       Mapped       Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1                   18619        26         0        1000           00:01.9





;                               COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:EXTDIR/OBJ=OBJ$:EXTDIR MSRC$:EXTDIR/UPDATE=(ENH$:EXTDIR)

; Size:          498 code + 0 data bytes
; Run Time:         00:12.3
; Elapsed Time:     00:37.7
; Lines/CPU Min:      3183
; Lexemes/CPU-Min: 13762
; Memory Used:  163 pages
; Compilation Complete
```

0165 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY